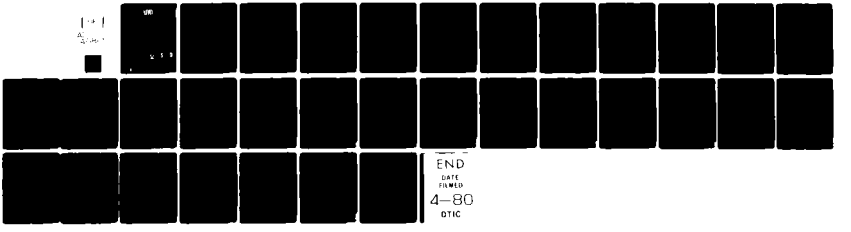
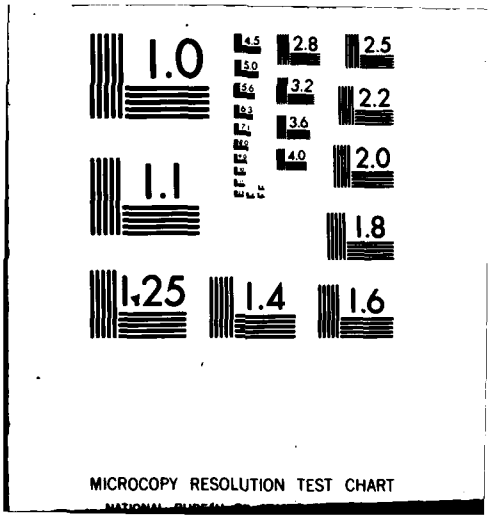
AD-A081 700 NAVAL RESEARCH LAB WASHINGTON DC F/G 9/4
A HEURISTIC METHOD OF OPTIMAL GENERALIZED HYPERCUBE ENCODING FO--ETC(U)
JAN 80 C C YANG, S K CHANG, K K SINGH
UNCLASSIFIED NRL-8382 SBIE-AD-E000 368 NL



END
DATE FILMED
4-80
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART

A Heuristic Method of Optimal Generalized Hyperscale Encoding for Pictorial Databases

C. C. Yang

Systems Research Branch
Space Systems Division

S. K. Chang and K. K. Sung

Department of Information Engineering
University of Ill.
Chicago, Illinois

January 15, 1980

NAVAL RESEARCH LABORATORY
Washington, D.C.

(14) NRL-8382

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NRL Report 8382
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): A HEURISTIC METHOD OF OPTIMAL GENERALIZED HYPERCUBE ENCODING FOR PICTORIAL DATABASES
5. TYPE OF REPORT & PERIOD COVERED: Interim report on a continuing NRL problem number
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): C.C. Yang, S.K. Chang, K.K. Singh*
8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Research Laboratory, Washington, DC 20375
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NRL Problem 79-0721-0-0

11. CONTROLLING OFFICE NAME AND ADDRESS
12. REPORT DATE: January [illegible] 1980
13. NUMBER OF PAGES: 33

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

*Present address: Department of Information Engineering, University of Illinois, Chicago, Illinois.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Generalized hypercube encoding
Updating a pictorial database
Threshold density
Similarity retrieval
Handle set
Optimal GH encoding

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Similarity retrieval from a large pictorial database can be much more efficient by encoding the the original database into certain convenient formats. Generalized hypercube (GH) encoding is one such technique. To optimize GH encoding, a heuristic approach has been formulated. Two optimization problems have been considered here: First, given the handle length m, find the optimal GH_m encoding. Second, given the threshold density, find the optimal GH encoding such that each generated hypercube has a density no less than a threshold density.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

A HEURISTIC METHOD OF OPTIMAL GENERALIZED HYPERCUBE ENCODING FOR PICTORIAL DATABASES

INTRODUCTION

In similarity retrieval from a pictorial database [1], pattern recognition [2,4], and clustering analysis, it is often desired to find the set of database records (or set of patterns, n-dimensional feature vectors, etc.) that are most similar to a test record (or test pattern, test feature vector, etc.). In the case of large databases (or patterns, clusters), it is very useful to encode the original database into certain convenient format in order to facilitate similarity retrieval and updating.

In this report, we shall consider one such technique called "Generalized Hypercube" encoding [3] and describe a heuristic method of generating minimum number of GH encoded tuples.

GENERALIZED HYPERCUBE ENCODING

Given a set of points $S = \{ p_1, p_2, \ldots, p_k \}$ in n-dimensional space, where $p_i = (x_1, x_2, \ldots, x_n)$ denotes a point in S, we can have a family of n-dimensional GH codes. For each m, $1 \leq m \leq n + 1$, the GH_m codes are:

$$(x_1, \ldots, x_{m-1}; a_m, \ldots, a_n; b_m, \ldots, b_n),$$

where $(x_1, x_2, \ldots, x_{m-1}, z_m, \ldots, z_n)$ is in S for some coordinates, $z_m, \ldots, z_n$; and
$a_j = \min \{ y_j$: for some coordinates, $z_k, (x_1, \ldots, x_{m-1}, z_m, \ldots, y_j, \ldots, z_n)$ in S$\}$,
$b_j = \max \{ y_j$: for some coordinates, $z_k, (x_1, \ldots, x_{m-1}, z_m, \ldots, y_j, \ldots, z_n)$ in S$\}$.

When m = 1, we are simply using the smallest n-dimensional hypercube containing S as the GH code. When m = n+1, the original point set S is used as the GH code. Other in-between values of m give GH codes of various levels of details.

For example, if point set $S = \{(1,1,3),(1,1,5),(1,2,1),(1,2,4),(2,3,6)\}$, GH_1 is $\{(1,1,1;2,3,6)\}$. GH_2 is $\{(1;1,1;2,5),(2;3,6;3,6)\}$. GH_3 is $\{(1,1;3;5),(1,2;1;4),(2,3;6;6)\}$. GH_4 is S itself. It should be noted that other GH codes can be generated, if we permute the coordinates. The previous example we have $\{(3;1,1;1,1),(5;1,1;,1,1),(1;1,2;1,2),(6;2,3;2,3)\}$.

Therefore, it is important to select carefully the coordinates on which the GH encoding technique can be applied, i.e. to generate a minimum number of GH encoded tuples the problem reduces to select a special "handle" set. It is easy to see that once a handle vector has been selected, the GH encoding is unique.

Manuscript submitted November 5, 1979.

In this report we shall describe the following heuristic methods for problems:

- Select a handle set $\{i_1, i_2, \ldots, i_{m-1}\}$ from $\{1, 2, \ldots, n\}$ so that the least number of GH encoded tuples are generated.

- Suppose we define the density of a hypercube H as the number of data points in H, divided by total number of lattice integers.). For example the hypercube (1,1,1;2,3,6) previously described has a density of $5/2 \cdot 3 \cdot 6 = 0.1388$. Intuitively, we would like to generate hypercubes having high densities. Therefore Problem 2 can be stated as follows:

Given a threshold density t, $0 \leq t \leq 1.0$, find optimal GH encoding such that each hypercube has density no less than t.

PROBLEM 1

Given m, the method of choosing m-1 handles which will generate optimal GHm codes for a point set S is based on the following ideas:

- For each handle (column) i, (i = 1, 2, . . ., n) consider vector $V_i = (x_{1i}, x_{2i}, \ldots, x_{ki})$ where k is the number of points in the set S. Among the elements $x_{i1}, x_{i2}, \ldots, x_{ik}$ those having the same value are grouped together and their associated points (in S) will consist of disjoint groups, say, $G_{1i}, G_{2i}, \ldots, G_{\ell i}$. Define the count (or cardinality) of $G_{ji} = C_j(i)$. We next compute a measure of the priority of the ith column p_i.

- $p_i = \sum_{j=1}^{\ell} [C_j(i)-1]^2$.

Intuitively, we ignore those groups having only one element and put heavier weight to those groups having more than one element. Obviously, there are many other ways to define a priority measurement for each handle.

- Based on the measure P_i, (i = 1, 2, . . ., n) choose the best m-1 handles, i.e., i's which have larger values of P_i.

Following is the algorithm to implement the above ideas:

ALGORITHM

Step 1.

To conpute P_i, consider vector $V_i = (x_{1i}, x_{2i}, \ldots, x_{ki})$,

a. Make a stack of all k elements of V_i.

b. Set a test element, TELMNT = top element of stack and set COUNT = 1.

c. Replace the top element of the stack by the last element and, set k = k-1; if $k < 1$, go to substep e.

d. Restack the k elements to check that the top element = TELMNT.

- If it does, COUNT = COUNT + 1; then go to substep c.
- If it does not, then

{ If COUNT > 1, then $P_i = P_i$ + COUNT * COUNT and COUNT = 1. TELMNT = top element of the stack; go to substep c }.

e. Include the last group's population count in P_i.

If COUNT > 1, then $P_i = P_i$ + COUNT * COUNT.

Step 2.

Using STEP 1, compute all P_i's for i = 1, 2, . . ., n. Store handle numbers in an array, such as [c(i) = i, i = 1, . . ., n].

Step 3.

Sort array P in descending order, and while moving the elements of P during sort, move elements of c also in the same fashion.

For generating GH_m codes, choose handles in the first (m - 1) positions of array c, which correspond to the best (m - 1) values of P_i, i = 1, 2, . . ., n.

Once the set of handles has been selected, the GH encoding is unique.

For example, let point set S = { (3,1,1), (5,1,1), (1,1,2), (4,1,2), (6,2,3) }. Using the previous algorithm, the measure of priority, P_i for i = 1,2,3 are: $P_1 = 0$, $P_2 = 16$, $P_3 = 8$. Based on the best values of P_i, we choose handle set (x_2,x_3,x_1) and with respect to this set of handles (GH_1) is { (1,1,1;2,3,6) }. GH_2 is { (1;1,1;2,5), (2;3,6;6) }. GH_3 is { (1,1;3;5), (1,2;1;4), (2,3;6;6) }. GH_4 is { (1,1,3), (1,1,5), (1,2,1), (1,2,4), (2,3,6) }.

EXPERIMENTAL RESULTS

The previous algorithm was implemented on an IBM/370 in FORTRAN; a listing of the program is given in Appendix A. The data for the program were random numbers with normal distribution and standard deviation, a number for each column vector V_i was also a random number with uniform distribution.

The selected set of handles by the previous method was used to generate GH_m encoded tuples, and their number was counted for a different number of point set and dimensions of space. To find the optimal GH_m encoding, great effort was taken, and the GH_m codes with minimum number of GH_m tuples was taken as the optimal GH_m encoding. (See Tables 1, 2, and 3.)

Few other heuristic approaches were tested to select the proper set of handles to generate optimal GH encoding, but the previous method has shown the best results and will be used in the next optimization problem.

PROBLEM 2

From the experimental results we know that the smaller the value of m (length of handles) the smaller the number of GH_m encoded tuples for a point set S, but at the same time such hypercubes tend to be sparse. Therefore, to optimize GH codes for a given

Table 1 — The number of optimal GH_m encoded tuples

No. of Points (M)	GH_2			GH_3		
	H	O	Error (%)	H	O	Error (%)
10	8	8	0.0	10	10	0.0
20	15	15	0.0	20	20	0.0
30	22	22	0.0	30	30	0.0
40	12	12	0.0	33	33	0.0
50	34	34	0.0	50	50	0.0
60	16	16	0.0	56	55	1.02
70	24	24	0.0	68	68	0.0
80	30	30	0.0	75	75	0.0
90	27	27	0.0	88	87	1.16
100	20	20	0.0	86	86	0.0

Dimension N = 3
H = heuristic
O = optimum

Table 2 — The number of optimal GH_m encoded tuples

No. of Points (M)	GH_2			GH_3			GH_4		
	H	O	Error (%)	H	O	Error (%)	H	O	Error (%)
10	8	8	0.0	10	10	0.0	10	10	0.0
20	16	16	0.0	20	20	0.0	20	20	0.0
30	10	10	0.0	29	28	3.57	30	30	0.0
40	21	21	0.0	40	39	2.56	40	40	0.0
50	16	16	0.0	44	44	0.0	50	50	0.0
60	25	25	0.0	59	59	0.0	60	60	0.0
70	28	28	0.0	68	68	0.0	70	70	0.0
80	21	21	0.0	74	71	4.23	80	80	0.0
90	18	18	0.0	75	75	0.0	90	90	0.0
100	22	22	0.0	93	93	0.0	100	100	0.0

Dimension N = 4
H = heuristic
O = optimum

Table 3 — The number of optimal GH_m encoded tuples

No. of Points (M)	GH_2			GH_3			GH_4			GH_5		
	H	O	Error (%)	H	O	Error (%)	H	O	Error (%)	H	O	Error (%)
10	8	8	0.0	10	10	0.0	10	10	0.0	10	10	0.0
20	13	13	0.0	20	19	5.26	20	20	0.0	20	20	0.0
30	10	10	0.0	29	29	0.0	30	30	0.0	30	30	0.0
40	16	16	0.0	38	38	0.0	40	40	0.0	40	40	0.0
50	21	21	0.0	49	49	0.0	50	50	0.0	50	50	0.0
60	18	18	0.0	55	55	0.0	59	59	0.0	60	60	0.0
70	18	18	0.0	64	63	1.58	70	70	0.0	70	70	0.0
80	21	21	0.0	75	75	0.0	80	79	1.26	80	80	0.0
90	26	26	0.0	84	84	0.0	90	90	0.0	90	90	0.0
100	22	22	0.0	96	92	4.35	100	100	0.0	100	100	0.0

Dimension N = 5
H = heuristic
O = optimum

threshold density t, find a set of handles with minimum length m, so that the resultant hypercubes have densities no less than theshold density, t.

Two special cases can be solved immediately. If t = 0, then we can select m = 1, so that a single n-dimensional hypercube is generated, and GH_1 becomes the optimum GH encoding. If t = 1.0, then all the hypercubes must have unit density, and we can select m = n + 1 so that the original point set S becomes the optimum GH encoding. For other values of t, m should fall between 1 and n + 1.

ALGORITHM

The algorithm optimize GH encoding for a given threshold density.

Step 1.

Start with m = 1;

generate GH_1 codes, which is one hypercube. If the density of this hypercube is > = threshold density, then this is the optimum GH encoding, and go to Step 7,

otherwise, set m = m + 1.

Step 2.

By using the previous heuristic method discussed in Problem 1, choose a set of handles which will generate the optimal GH_m encoding for a given m.

Set a flag = 0; and

NUM = 0, where NUM will be the number of GH_m encoded tuples with respect to the chosen handle vector of length m.

Step 3.

a. Generate the next hypercube with respect to the above chosen handle vector and compute its density.

b. If density of this hypercube = > threshold density, and

if the number of points in this hypercube > 1, then

{ include this hypercube in optimal GH encoding.

NUM = NUM + 1; and

delete the points included in this hypercube from the original point set. }

c. If the density of this hypercube < threshold density, then set the flat = 1

Step 4.

Repeat Step 3 until all the points in point set have been considered once.

Step 5.

If the number of points in remaining point set = 0 then go to Step 7.

If (number of points in point set = 1 or

NUM < = 0 and flag = 0), then go to Step 6.

If (NUM < = 0 and flag = 1), then m = m + 1.

Go to Step 2.

Step 6.

Generate GH_m encoded tuples with respect to the last handle vector of length m. Each GH_m encoded tuple is included in the optimal GH encoding.

Step 7.

Stop.

As an illustration, if point set S = { (3,1,1),(5,1,1),(1,1,2),(4,1,2),(6,2,3) } and threshold density = 50% (0.5), then with the previous algorithm, optimal encoding is attended as {(1;1,2;2,5), (2;3,6;3,6) } and the density of each hypercube is > = 50%.

EXPERIMENTAL RESULTS

The previous algorithm also was implemented on an IBM/370 in FORTRAN. A listing of the program is given in Appendix B. The test data were again random numbers.

The results are in Tables 4, 5, and 6.

DISCUSSION

In this section, some interesting applications of the generalized GH encoding technique are discussed.

Reference 1 suggests that a relational file R can be characterized by a logical expression E, so that every n-tuple $(x_1, x_2, \ldots, x_n)$ in R satisfies this logical expression.

The generalized GH encoding technique can be used to find a suitable logical expression characterizing a relational file.

For example, if S is as given in the Generalized Hypercube Encoding section of this report and m = 1, then GH_1 = { (1,1,1;2,3,6) } and the corresponding logical expression is $(1 <= x_1)$ & $(1 <= x_2)$ & $(1 <= x_3)$ & $(x_1 <= 2)$ & $(x_2 <= 3)$ & $(x_3 <= 6)$;

when m = 2, then GH_2 = { (1;1,1;2,5),(2;3,6;3,6) } and the corresponding logical expression is $(x_1 = 1)$ & $(1 <= x_2)$ & $(1 <= x_3)$ & $(x_3 <= 5))$ V $((x_1 = 2)$ & $(x_2 = 3)$ & $(x_3 = 6))$, thus, each GH encoding corresponds to a logical characteristic expression for the relational file.

Table 4 — The number of optimal GH encoded tuples

(M)	t=10	t=20	t=30	t=40	t=50	t=60	t=70	t=80	t=90
10	6	7	7	7	9	9	10	10	10
20	11	14	18	18	18	18	18	18	18
30	9	23	24	24	24	26	28	28	28
40	18	36	36	36	36	37	38	38	38
50	23	36	39	40	42	48	50	50	50
60	12	43	43	48	53	55	58	58	58
70	21	40	46	46	49	52	64	64	64
80	1	52	53	54	57	57	60	60	60
90	1	45	50	52	56	77	80	84	84
100	1	40	52	58	60	81	82	91	91

M = number of points in point set
t = percent threshold density
Dimension N = 3

Table 5 — The number of optimal GH encoded tuples

(M)	t=10	t=20	t=30	t=40	t=50	t=60	t=70	t=80	t=90
10	7	7	9	9	9	9	10	10	10
20	14	17	17	18	18	18	18	18	18
30	25	28	28	29	29	29	29	29	29
40	32	36	39	39	39	39	39	39	39
50	38	45	46	49	50	50	50	50	50
60	46	52	56	57	57	57	59	59	59
70	42	57	60	68	68	68	70	70	70
80	45	72	77	78	79	79	79	79	79
90	58	72	78	79	82	87	88	88	88
100	66	78	81	90	91	92	92	93	93

M = number of points in point set
t = percent threshold density
Dimension N = 4

Table 6 — The number of optimal GH encoded tuples

(M)	t=10	t=20	t=30	t=40	t=50	t=60	t=70	t=80	t=90
10	10	10	10	10	10	10	10	10	10
20	20	20	20	20	20	20	20	20	20
30	28	29	30	30	30	30	30	30	30
40	40	40	40	40	40	40	40	40	40
50	47	50	50	50	50	50	50	50	50
60	57	58	58	58	58	59	59	59	59
70	68	68	69	69	69	70	70	70	70
80	76	77	77	77	77	79	79	79	79
90	79	86	87	87	88	89	90	90	90
100	95	100	100	100	100	100	100	100	100

M = number of points in point
t = percent threshold density
Dimension N = 5

In addition to applications in database characterization, the previously described technique also can be applied in clustering analysis where clusters are described by hypercubes with handles. Thus optimization technique will generate optimal encoding for the clusters.

A third application can be the description of n-dimensional pictures, which will have a practical significance in computer graphics and image processing applications.

REFERENCES

1. S.K. Chang et al. "Design Considerations of a Pictorial Database System," International Journal of Policy Analysis and Information Systems 1 (No. 2) (Jan. 1978).

2. K.S. Fu, *Syntectic Methods in Pattern Recognition*, Academic Press, 1973, pp 69-82.

3. C.C. Yang and S.K. Chang, "Encoding Techniques for Efficient Similarity Retrieval From Pictorial Databases", in *Pictorial Information Systems*, Springer-Verlag, 1979.

4. T. Pavlidis, *Structural Pattern Recognition*, Springer-Verlag, 1977.

APPENDIX A

```
//JUNK JOB
/*JOBPARM T=(8,0),L=2,R=256
// EXEC FORTGCLG
//FORT.SYSIN DD *
CCCCCCCCCC
C
C    MAIN PROGRAM TO GENERATE OPTIMAL GHM ENCODING FOR A POINTSET BY
C    A HEURISTIC METHOD & TO CHECK THE RESULTS. NUMBER OF THESE GHM
C    ENCODED TUPLES IS COMPARED WITH THE MINIMUM (REAL OPTIMUM) NO.
C    OF GHM ENCODED TUPLES FOUND IN EXHUASTIVE APPROACH.
C
C    CALLS:
C      (1) GENPTS: TO GENERATE TEST DATA I.E. A POINT SET.
C      (2) HEURIS: TO GENERATE OPTIMAL GHM ENCODED TUPLES BY A
C            HEURISTIC METHOD.
C      (3) EXHAUS: TO GENERATE OPTIMAL GHM ENCODING BY EXHAUSTIVE
C            APPROACH.
C
CCCCCCCCCC
      INTEGER PTSET(110,6),P(6),NCODE(5,3)
      IFL=1
      N = 4
      M = 50
      CALL GENPTS(PTSET,M,N)
      I=(IFL .EQ. 1) GO TO 440
      PRINT 1
1     FORMAT('1',10X, 'INITIAIL POINT SET IS:')
      DO 100 I = 1,M
      PRINT 2 ,(PTSET(I,J),J=1,N)
100   CONTINUE
2     FORMAT('0',10X,'(',5I3,')')
      PRINT 3
3     FORMAT(////,100('*'))
      PRINT 4
4     FORMAT(//,25X,'HEURISTIC APPROACH')
      PRINT 5
5     FORMAT(25X,'--------- --------')
      PRINT 6
6     FORMAT(//,100('*'))
440   CALL HEURIS(PTSET,M,N,NCODE,IFL)
      I=(IFL .EQ. 1) GO TO 442
      PRINT 11
11    FORMAT(///,100('*'))
      PRINT 12
12    FORMAT(//,21X,'EXHAUSTIC APPROACH')
      PRINT 13
13    FORMAT(21X,'--------- --------')
      PRINT 14
14    FORMAT(//,100('*'))
442   CALL EXHAUS(PTSET,M,N,NCODE,IFL)
      N1 = N-1
      PRINT 20,M
20    FORMAT(//,10X,'NO. OF POINTS IS = ', I4)
      PRINT 22,N
22    FORMAT(10X,'NO. OF DIMENSINS = ', I3)
      PRINT 15
15    FORMAT(//,20X,'HEURISTIC',10X,'OPTIMUM',8X,'WORST',8X,'%ERR')
      DO 200 I = 1,N1
      ERR = (NCODE(I,1) - NCODE(I,2))*100./NCODE(I,2)
200   PRINT 16, I, (NCODE(I,J),J= 1,3),ERR
```

```
      16     FORMAT('0',10X,'GH',I2,3(10X,I4),F16.3)
             IF(IFL .EQ. 1) PRINT 66
      66     FORMAT(//,100('.'))
     500     CONTINUE
     600     CONTINUE
             STOP
             END
CCCCCCCCCC
C
C    RANDOM(X): GENERATES A RANDOM NUMBER X , 0<= X <=1.0 WITH
C               UNIFORM DISTRIBUTION.
C
CCCCCCCCCC
      SUBROUTINE RANDOM(X)
      REAL X
      INTEGER A/19727/, MULT/25211/, BASE/32768/
      A = MOD(MULT*A,BASE)
      X = FLOAT(A)/FLOAT(BASE)
      RETURN
      END
CCCCCCCCCC
C
C    GENPTS(PTSET,M,N):- TO GENERATE TEST DATA SET IN ARRAY PTSET,
C        WHICH CONTAINS M NUMBER OF POINTS IN N DIMENSIONAL SPACE.
C    CALLS:
C       (1) FA034: A LIBRARY SUBROUTINE TO GENERATE RANDOM NUMBERS OF
C                  NORMAL DISTRIBUTION WITH A GIVEN STANDARD
C                  DEVIATION.
C       (2) RANDOM: TO GET ANOTHER RANDOM NUMBER TO BE USED AS FIX
C                  STANDARD DEVIATION FOR ONE COLUMN.
C
CCCCCCCCCC
      SUBROUTINE GENPTS(PTSET, M,N)
      INTEGER PTSET(110,6)
      DO 100 J = 1, N
       CALL RANDOM(U)
       STD = 19.0*U + 3
       DO 100 I = 1, M
        CALL FA034(STD,X)
        PTSET(I,J) = INT(X + .5)
  100 CONTINUE
      RETURN
      END
CCCCCCCCCC
C
C    PROCES(PTSET,P,M,N):- COMPUTES THE MEASURE OF PRIORITY P(I)
C         FOR EACH COLUMN I WHERE I = 1,...,N FOR A GIVEN POINT SET
C         IN ARRAY PTSET CONTAINING M POINTS IN N DIMENSIONAL SPACE.
C    CALLS:
C       (1) HPODER: TO COMPUTE P(I) FOR ONE PARTICULAR VALUE OF I.
C
CCCCCCCCCC
      SUBROUTINE PROCES(PTSET,P,M,N)
      INTEGER PTSET(110,6),TEMPR(110),P(6)
      DO 100 I = 1,N
      DO 200 J = 1,M
  200 TEMPR(J) = PTSET(J,I)
      CALL HPODER(1,TEMPR,M,P)
  100 CONTINUE
      RETURN
```

```
      END
CCCCCCCCCC
C
C   HPODER(I,TEMPR,M,P):- COMPUTES P(I) FOR A COLUMN VECTOR TEMPR
C       AND USES HEAP STRUCTURE.
C   CALLS:
C     (1) REHEAP: TO MAKE HEAP OF ELEMENTS OF ARRAY TEMPR.
C
CCCCCCCCCC
      SUBROUTINE HPODER(I,TEMPR,M,P)
      INTEGER TEMPR(110),P(6)
      INTEGER TEMP,ELMENT,PSUM,SUM
       MI = M
       L = M/2 + 1
10     L = L-1
       CALL REHEAP(L,MI,TEMPR)
       IF (L .GT. 1) GOTO 10
       ELMENT = 0
       PSUM = 0
       SUM = 0
20     TEMP = TEMPR(1)
       TEMPR(1) = TEMPR(MI)
       IF (ELMENT .EQ. TEMP)GO TO 30
        IF(SUM .GT. 1) PSUM = PSUM + SUM * SUM
        ELMENT = TEMP
        SUM = 1
        GO TO 40
30     SUM = SUM + 1
40     MI = MI - 1
       IF(MI .GT. 0) GO TO 50
         IF (SUM .GT. 1) PSUM = PSUM + SUM *SUM
         P(I) = PSUM
         RETURN
50     CALL REHEAP(1,MI,TEMPR)
       GOTO 20
       END
CCCCCCCCCC
C
C  REHEAP(L,MI,TEMPR):- MAKES HEAP OF ELEMENTS OF TEMPR STARTING
C       FROM TEMPR(L) TO TEMPR(MI).
C
CCCCCCCCCC
      SUBROUTINE REHEAP(L,MI,TEMPR)
      INTEGER TEMPR(110)
      INTEGER FLAG , X
      II = L
      JJ = 2*II
      X = TEMPR(II)
      FLAG = 1
      IF (JJ .GT. MI) RETURN
40    IF(JJ .GE. MI) GOTO 10
      IF (TEMPR(JJ) .LT. TEMPR(JJ+1)) JJ =JJ+1
10    IF(X .GE. TEMPR(JJ)) GOTO 20
      TEMPR(II)  =  TEMPR(JJ)
      II = JJ
      JJ = 2*II
      TEMPR(II) = X
      GOTO 30
20    FLAG = 0
30    IF (JJ .LE. MI .AND. FLAG .EQ. 1) GOTO 40
```

```
      RETURN
      END
CCCCCCCCCC
C
C  SORT(P,O,N):- SORTS P ARRAY CONTAINING N ELEMENTS AND DURING
C       THIS SORT ALSO MOVES THE ELEMENTS OF ARRAY O IN THE SAME
C       FASHION.
C
CCCCCCCCCC
      SUBROUTINE SORT(P,O,N)
      INTEGER O(6),P(6)
      INTEGER FLAG, TOP, TEMP, BOTTOM
      FLAG = 1
      BOTTOM = N
      TOP = 1
10    IF(FLAG .NE. 0 .AND. TOP .LT. BOTTOM) GO TO 20
      RETURN
20    FLAG = 0
      I1 = BOTTOM -1
      DO 100 I = 1,I1
      IF (P(I) .GE. P(I+1)) GO TO 100
          X  = P(I)
          P(I) =P(I+1)
          P(I+1) = X
          TEMP =O(I)
          O(I) =O(I+1)
          O(I+1)=TEMP
          FLAG = 1
100   CONTINUE
      BOTTOM = BOTTOM -1
      GO TO 10
      END
CCCCCCCCCC
C
C  SORTPT(PTSET,O,M,N):- TO SORT POINTS OF POINTSET IN PTSET W.R.T.
C     HANDLE VECTOR IN ARRAY O. HEAP SORT IS USED FOR THIS.
C  CALLS:
C    (1) REHIP: TO MAKE HEAP.
C    (2) COMPAR: TO COMPARE ORDER OF TWO POINTS.
C    (3) EXCHAN: TO EXCHANGE POSITIONS OF TWO POINTS IN PTSET.
C
CCCCCCCCCC
      SUBROUTINE SORTPT(PTSET,O,M,N)
      INTEGER PTSET(110,6),O(6)
      MI = M
      L = MI/2 + 1
10    L = L-1
      CALL REHIP(L,MI,M,N,O,PTSET)
      IF(L .GT.1) GOTO 10
20    CALL EXCHAN(1,MI,N,PTSET)
      MI = MI-1
      CALL REHIP(1,MI,M,N,O,PTSET)
      IF(MI .GT. 1) GO TO 20
      RETURN
      END
CCCCCCCCCC
C
C   REHIP(L,MI,M,N,O,PTSET):- MAKES A HEAP OF ELEMENTS OF PTSET
C       STARTING FROM L TO MI. PTSET CONTAINS M POINTS IN N
C       DIMENSIONAL SPACE.
```

```
241.      C
242.      CCCCCCCCCC
243.            SUBROUTINE REMIP(L,M1,M,N,D,PTSET)
244.            INTEGER PTSET(110,6),D(6)
245.            INTEGER FLAG ,X
246.            I = L
247.            J = 2*I
248.            X = M+1
249.            DO 5 LL=1,N
250.      5     PTSET(X,LL) = PTSET(I,LL)
251.            FLAG = 1
252.            IF(J .GT.M1) RETURN
253.      40    IF(J .GE. M1) GO TO 10
254.            CALL COMPAR(J,J+1,N,D,PTSET,ICOMP)
255.            IF (ICOMP .LT.0) J = J+1
256.      10    CALL COMPAR(X,J,N,D,PTSET,ICOMP)
257.            IF (ICOMP .GE.0) GO TO 20
258.            DO 100 LL = 1,N
259.      100   PTSET(I,LL) = PTSET(J,LL)
260.            I = J
261.            J = 2*I
262.            DO 200 LL = 1,N
263.      200   PTSET(I,LL) = PTSET(X,LL)
264.            GO TO 30
265.      20    FLAG = 0
266.      30    IF (J .LE. M1 .AND. FLAG .EQ.1) GO TO 40
267.            RETURN
268.            END
269.      CCCCCCCCCC
270.      C
271.      C    COMPAR(I,J,N,D,PTSET,ICOMP):- COMPARES TWO POINTS I & J W.R.T.
  2.      C       A HANDLE VECTOR IN D, AND RETURNS IN ICOMP:
  3.      C          1 IF ITH POINT > JTH POINT.
274.      C          0 IF ITH POINT = JTH POINT.
275.      C         -1 IF ITH POINT < JTH POINT.
276.      C
277.      CCCCCCCCCC
278.            SUBROUTINE COMPAR(I,J,N,D,PTSET,ICOMP)
279.            INTEGER PTSET(110,6),D(6)
280.            II = 1
281.      100   III = D(II)
282.            IF (PTSET(I,III) .NE. PTSET(J,III)) GO TO 10
283.            IF (II .LT. N) GO TO 5
284.            ICOMP = 0
285.            RETURN
286.      5     II = II+1
287.            GO TO 100
288.      10    IF(PTSET(I,III) .LT. PTSET(J,III)) ICOMP = -1
289.            IF (PTSET(I,III).GT.PTSET(J,III)) ICOMP = 1
290.            RETURN
291.            END
292.      CCCCCCCCCC
293.      C
294.      C     EXCHAN(I,J,N,PTSET):-SWITCHES THE POSITIONS OF TWO POINTS I &
295.      C        J IN ARRAY PTSET WHICH CONTAINS A POINT SET IN N DIMENSIONAL
296.      C        SPACE.
297.      C
298.      CCCCCCCCCC
299.            SUBROUTINE EXCHAN(I,J,N,PTSET)
300.            INTEGER PTSET(110,6)
```

```
      DO 100 II = 1,N
      ITEMP = PTSET(I,II)
      PTSET(I,II) = PTSET(J,II)
      PTSET(J,II) = ITEMP
100   CONTINUE
      RETURN
      END
CCCCCCCCCC
C
C    ENCODE(MM,M,N,O,PTSET,NUM,IFL):-GENERATES GHMM CODES FOR A
C      SET PTSET, CONTAINING M POINTS IN N DIMENSIONAL SPACE.
C      NUM WILL CONTAIN THE NUMBER OF SUCH GHMM ENCODED TUPLES IN IT.
C      IFL IS A FLAG TO TURN OFF AND ON THE PRINTING OF INTERMEDIATE
C      RESULTS.
C    CALLS:
C      (1) MINMAX: TO FIND THE MINIMUM & MAXIMUM ELEMENT IN A
C            PARTICULAR COLUMN FOR SOME POINTS CONSIDERED TO BE IN
C            CURRENT HYPERCUBE.
C
CCCCCCCCCC
      SUBROUTINE ENCODE(MM,M,N,O,PTSET,NUM,IFL)
      INTEGER PTSET(110,6),O(6),CODE(12)
      IF(IFL .EQ.1) GO TO 44
      PRINT 1
1     FORMAT(//)
      PRINT 5,MM
5      FORMAT(11X,'GH',I2,' CODES ARE:')
44    IF (MM .EQ. 1) GO TO 10
      IF(MM .EQ. N+1) GO TO 20
      NUM = 0
      N1 = MM-1
      M1 = M + 1
      I = 1
      J = 1
      DO 100 II = 2,M1
        IF (II .GT. M) GO TO 40
        FLAG = 0
        DO 200 JI = 1,N1
          JJ = O(JI)
          IF(PTSET(II-1,JJ) .EQ. PTSET(II,JJ)) GO TO 200
          FLAG = 1
          GO TO 30
200     CONTINUE
30      IF (FLAG .EQ. 1) GO TO 40
        J = J + 1
        GO TO 100
40      DO 300 JI = 1,N1
        JJ = O(JI)
        CODE(JI) = PTSET(I,JJ)
300     CONTINUE
        L = N1
        NN = N - N1
        DO 400 JI = MM,N
        JJ = O(JI)
        CALL MINMAX(PTSET,M,JJ,I,J,MIN,MAX)
        L = L + 1
        CODE(L) = MIN
400     CODE(L+NN) = MAX
        N2 = L
         N3 = L + NN
```

```
      IF(IFL .EQ. 1) GO TO 85
       PRINT 11,(CODE(I),I=1,N1)
11     FORMAT('0',10X,'(',3I3)
       PRINT 13,(CODE(I),I =MM,N2)
13     FORMAT('+',22X,':',3I3)
       N22 = N2+1
       PRINT 15,(CODE(I),I= N22,N3)
15     FORMAT('+', 33X,':',3I3)
       PRINT 16
16     FORMAT('+', 44X,')')
85     I = II
       J = II
       NUM = NUM + 1
100    CONTINUE
       RETURN
10     DO 111 I = 1,N
       JJ = O(I)
       CALL MINMAX(PTSET,M,JJ,1,M,MIN,MAX)
       CODE(I) = MIN
       CODE(I+N) = MAX
111    CONTINUE
       IF(IFL .EQ. 1) RETURN
       PRINT 8,(CODE(I),I=1,N)
8      FORMAT('0',10X,'(',4I3)
       NN = N + 1
       N2 = 2*N
       PRINT 90,(CODE(I),I=NN,N2)
90     FORMAT('+', 24X,' :',4I3)
       PRINT 93
93     FORMAT('+', 38X,' )')
       RETURN
20     DO 222 I = 1 , M
        DO 220 J = 1 , N
         JJ = O(J)
220    CODE(J) = PTSET(I,JJ)
       IF (IFL .EQ. 1) GO TO 222
       PRINT 22,(CODE(J),J = 1,N)
22     FORMAT('0',10X,'(',4I3)
       PRINT 23
23     FORMAT('+', 24X,' )')
222    CONTINUE
       RETURN
       END
CCCCCCCCCC
C
C    MINMAX(PTSET,M,ICOL,I,J,MIN,MAX):- TO FIND THE MINIMUM &
C       MAXIMUM ELEMENTS OF COLUMN ICOL,STARTING FORM I TO J IN ARRAY
C       PTSET.
C
CCCCCCCCCC
      SUBROUTINE MINMAX(PTSET,M,ICOL,I,J,MIN,MAX)
      INTEGER PTSET(110,6)
      MIN = PTSET(I,ICOL)
      MAX = PTSET(I,ICOL)
      DO 100 II = I,J
         IF (PTSET(II,ICOL) .LT. MIN) MIN = PTSET(II,ICOL)
         IF (PTSET(II,ICOL) .GT. MAX) MAX = PTSET(II,ICOL)
100   CONTINUE
      RETURN
      END
```

```
421.      CCCCCCCCCC
422.      C
423.      C    HEURIS(PTSET,M,N,NCODE,IFL):- TO STORE THE NUMBER OF GHI ENCODED
424.      C        TUPLES FOR I=1,2,...,N+1 IN ARRAY NCODE. THE SET OF HANDLES
425.      C        IS CHOOSEN BY HEYRISTIC METHOD.
426.      C    CALLS:
427.      C      (1) PROCES: TO COMPUTE THE MEASURE OF PRIORITY P(I) FOR EACH
428.      C            COLUMN I.
429.      C      (2) SORT: TO GET THE PROPER SET OF HANDLES CORRESPONDING  TO
430.      C            TO BEST VALUES OF THEIR PRIORITY P(I).
431.      C      (3) SORTPT: TO SORT POINT SET PTSET W.R.T. CHOOSEN SET OF
432.      C            HANDLES.
433.      C      (4) ENCODE: TO GENERATE GH ENCODED TUPLES.
434.      C
435.      CCCCCCCCCC
436.            SUBROUTINE HEURIS(PTSET,M,N,NCODE,IFL)
437.            INTEGER PTSET(110,6),D(6),P(6),NCODE(5,3)
438.            CALL PROCES(PTSET,P,M,N)
439.            PRINT 1
440.      1     FORMAT(///, 11X,'PRINTOUT OF ARRAY P IS:')
441.            PRINT 2 ,(P(I),I=1,N)
442.      2     FORMAT(10X,5I10)
443.            DO 200 I = 1,N
444.      200   D(I)  = I
445.            CALL SORT(P,D,N)
446.            PRINT 3
447.      3     FORMAT('0',10X,'HANDLES CHOSEN ARE:')
448.            PRINT 4  ,(D(I),I=1,N)
449.      4     FORMAT(' ',10X,4I5)
450.            CALL SORTPT(PTSET,D,M,N)
 61.            IF(IFL .EQ. 1) GO TO 44
 62.            PRINT 5
453.      5     FORMAT('0',10X,'SORTED POINT SET IS:')
454.            DO 300 I = 1,M
455.      300   PRINT 10,(PTSET(I,J),J= 1,N)
456.      10    FORMAT(11X,'(',4I3,')')
457.            PRINT 11
458.      11    FORMAT(///,20X,'FOLLOWING ARE GH CODES W.R.T.ABOVE HANDLES:')
459.      44    N1 = N + 1
460.            DO 400 I = 1,N1
461.            CALL ENCODE(I,M,N,D,PTSET,NUM,IFL)
462.            IF( I .LE.1 .OR. I .GE. N1) GO TO 400
463.              NCODE(I-1,1) = NUM
464.              NCODE(I-1,2) = NUM
465.              NCODE(I-1,3) = NUM
466.      400   CONTINUE
467.            RETURN
468.            END
469.      CCCCCCCCCC
470.      C
471.      C    EXHAUS(PTSET,M,N,NCODE,IFL):- TO STORE THE MINIMUM NUMBER OF GHI
472.      C        ENCODED TUPLES FOR I=1,2,...,N+1  IN ARRAY NCODE.ALL THE
473.      C        POSSIBLE SET OF HANDLES ARE USED TO GENERATE GH CODES & THE
474.      C        ONE WITH  MINIMUM NUMBER OF TUPLE IS TAKEN AS OPTIMAL GH
475.      C        CODES.
476.      C    CALLS:
477.      C      (1) RCOMB: TO FIN ALL POSSIBLE SETS OF HANDLES.
478.      C      (2) SORTPT: TO SORT THE POINTSET W.R.T. CURRENT SET OF HANDLES
479.      C      (3) ENCODE: TO GENERATE GH ENCODED TUPLES.
480.      C
```

```
      20          RETURN
      10          DO 222 I = 1,N
                     JJ = O(I)
                     CALL MINMAX(PTSET,ORDER,JJ,1,NPTS,MIN,MAX)
                     CODE(I) = MIN
                     CODE(I+N) = MAX
      222         CONTINUE
                  DEN = DENSTY(CODE,MM,N,1,NPTS)
                  IF(DEN .GE. THRESH) GOTO 444
                     NUM = 0
                     RETURN
      444         IF (FLG .EQ. 1) GO TO 666
                     PRINT 21,(CODE(I),I=1,N)
      21             FORMAT('0',10X,'(',6I3)
                     N1 = N+1
                     N2 = N+N
                     PRINT 22,(CODE(I),I=N1,N2)
      22             FORMAT('+',30X,',',6I3)
                     PRINT 23
      23             FORMAT('+',50X,')')
      666            NUM = 1
                     RETURN
                  END
      CCCCCCCCC
      C
      C   MINMAX(PTSET,ORDER,JJ,I,J,MIN,MAX):- FIND THE MINIMUN & MAXIMUM
      C       OF ELEMENTS OF COLUMN JJ ,STORED IN PTSET,THEIR POINTERS ARE
      C       STORED IN ARRAY ORDER STARTING FROM I TO J.
      C
      CCCCCCCCCC
               SUBROUTINE MINMAX(PTSET,ORDER,JJ,I,J,MIN,MAX)
               INTEGER PTSET(110,6),ORDER(110)
               II = ORDER(I)
               MIN = PTSET(II,JJ)
               MAX = PTSET(II,JJ)
               DO 100 K = I,J
                  II = ORDER(K)
                  IF(PTSET(II,JJ) .LT. MIN) MIN = PTSET(II,JJ)
                  IF( PTSET(II,JJ) .GT. MAX) MAX= PTSET(II,JJ)
      100      CONTINUE
               RETURN
               END
      CCCCCCCCCC
      C
      C    DENSTY(CODE(MM,I,J):- COMPUTES THE DENSITY OF HYPERCUBE
      C       GENERATED BY GHM ENCODED TUPLE STORED IN CODE WHICH INCLUDES
      C       POINTS POINTED TO BY POINTERS IN ORDER STARTING FROM I TO J.
      C
      CCCCCCCCCC
               FUNCTION DENSTY(CODE,MM,N,I,J)
               INTEGER CODE(12)
               NUMPTS = J-I +1
               VOL = 1
               M1 = MM + 1
               DO 100 K = M1 ,N
               VOL = VOL*(CODE(K+N-MM)-CODE(K)+1)
      100      CONTINUE
               DENSTY = NUMPTS / VOL
               RETURN
               END
```

```
541.                    L  = L + ITERM
542.                    II = II + 1
543.                    L1 = L - 1
544.                    IF (II .GT. L1) GO TO 20
545.                    DO 500 I2 = II , L1
546.                    TABLE(I2,COL) = TABLE(II,COL)
547.      500           CONTINUE
548.      20            IF (I .GT.N1) GO TO 400
549.                    CALL FILL(TABLE,COL,N,II,L)
550.      400           CONTINUE
551.      300           CONTINUE
552.                    K = NUM*K
553.                    NUM = NUM - 1
554.                    COL = COL + 1
555.                    IF (NUM .GE.2) GO TO 10
556.                    RETURN
557.                    END
558.      CCCCCCCCCC
559.      C
560.      C     FILL(TABLE,COL,N,II,L):- TO STORE SOME ELEMENTS IN LTH ROW OF
561.      C        ARRAY TABLE.
562.      C
  3.      CCCCCCCCCC
  4.                    SUBROUTINE FILL(TABLE,COL,N,II,L)
565.                    INTEGER TABLE(720,6)
565.                    INTEGER COL, COL1
567.                      COL1 = COL + 1
568.                      IX = TABLE(II,COL)
569.                      DO 100 I = COL1,N
570.      100             TABLE(L,I-1) = TABLE(II,I)
571.                      TABLE(L,N) = IX
572.                      RETURN
573.                      END
```

APPENDIX B

```
      /*JOBPARM T=(2,0),L=4
      // EXEC FORTGCLG
      //FORT.SYSIN DD *
      CCCCCCCCCCC
      C
      C          MAIN PROGRAM TO OPTIMIZE GH ENCODING FOR A POINTSET.
      C          SUCH THAT EACH HYPERCUBE HAS DENSITY NO LESS THAN A GIVEN
      C          THRESHOLD DENSITY.
      C  PTSET:          ARRAY CONTAINING POINTSET.
      C  THRESH:         THRESHOLD DENSITY.
      C  M:              NO.OF POINTS IN POINT SET.
      C  N:              DIMENSIONAL SPACE.
      C  FLG:            FLAG USED TO TURN OFF & ON PRINTING OF INTERMEDIATE
      C                  RESULTS.
      C  CALLS :-
      C    (1) GENPTS: TO GENERATE M POINTS IN N DIMENSIONAL SPACE &
      C                STORE THEM IN ARRAY PTSET.
      C    (2) OPTMUM: TO GENERATE OPTIMAL GH ENCODED TUPLES, AND
      C                RETURNS THEIR NUMBER IN NCODE. IF FLG IS ON ,PRINTS
      C                THOSE GH ENCODED TUPLES ALSO. THERSH IS THE THRESHOLD
      C                DENSITY SUCH THAT EACH HYPERCUBE HAS DENSITY NO LESS
      C                THAN THRESH.
      C
      CCCCCCCCCCC
            INTEGER PTSET(110,6),TABLE(11)
            INTEGER FLG
            N =  3
            FLG = 1
            DO 300 M = 10,100,10
            CALL GENPTS(PTSET,M,N)
            DO 400 ITH = 10,110,10
            NDEN = ITH - 10
            THRESH = NDEN/100.0
            IF (FLG .EQ. 1) GO TO 10
            PRINT 1
      1     FORMAT('1',10X, 'INITIAIL POINT SET IS:')
            DO 100 I = 1,M
            PRINT 2 ,(PTSET(I,J),J=1,N)
      100   CONTINUE
      2     FORMAT('0',10X,'(',6I3,')')
            PRINT 3
      3     FORMAT(////,100('*'))
            PRINT 4
      4     FORMAT(//,25X,'HEURISTIC APPROACH')
            PRINT 5
      5     FORMAT(25X,'--------- --------')
      10    CALL OPTMUM(PTSET,M,N,THRESH,NCODE,FLG)
      400   TABLE(ITH/10) = NCODE
            PRINT 6,N
      6     FORMAT(///,10X,'DIMENSION = ',I5)
            PRINT 7,M
      7     FORMAT(10X,'NO. OF POINTS IN SET= ',I5)
            PRINT 8
      8     FORMAT('0',10X,'THRESHOLD ',10X,'# OF CODES GENERATED')
            DO 200 I = 1,11
            ITH  = I * 10 - 10
               PRINT 9, ITH,TABLE(I)
      200   CONTINUE
      300   CONTINUE
      9     FORMAT(10X,I10,10X,I10)
```

```
      STOP
      END
CCCCCCCCCCC
C
C     RANDOM(X): GENERATE A RANDOM NUMBER X, 0 <= X <=1.0, WITH
C                UNIFORM DISTRIBUTION.
C
CCCCCCCCCCC
      SUBROUTINE RANDOM(X)
      REAL X
      INTEGER A/19727/, MULT/25211/, BASE/32768/
      A = MOD(MULT*A,BASE)
      X = FLOAT(A)/FLOAT(BASE)
      RETURN
      END
CCCCCCCCCC
C
C     GENPTS(PTSET,M,N):-  TO GENERATE TEST DATA SET IN ARRAY
C           PTSET, WHICH CONTAINS M NUMBERS IN N DIMENSIONAL
C           SPACE.
C     CALLS:
C       (1) RANDOM: TO GET A RANDOM NUMBER.
C
CCCCCCCCCC
      SUBROUTINE GENPTS(PTSET, M,N)
      INTEGER PTSET(110,6)
      DO 100 J = 1, N
       DO 100 I = 1, M
        CALL RANDOM(U)
        PTSET(I,J) = INT(9 * U) + 1
100   CONTINUE
      RETURN
      END
CCCCCCCCCC
C
C     OPTMUM(PTSET,M,N,THRESH,NCODE,FLG):-
C           TO GENERATE OPTIMAL GH ENCODING FOR A GIVEN THRESHOLD
C           DENSITY, THRESH; AND RETURNS THE NUMBER OF SUCH TUPLES
C           IN NCODE.
C  CALLS:-
C    (1) PROCES: TO COMPUTE THE MEASURE OF PRIORITY P(I) FOR
C           EACH COLUMN I.
C    (2) SORT:   TO GET THE PROPER SET OF HANDLES CORRESPONDING
C           TO BEST VALUES OF THEIR PRIORITIES, P(I).
C    (3) SORTPT:   TO SORT POINTSET IN PTSET A.R.T. NEW SET OF
C           HANDLES.
C    (4) ENCODE:   TO GENERATE GHM ENCODED TUPLES WITH DENSITY
C           NO LESS THAN THRESHOLD DENSITY.
C
CCCCCCCCCC
      SUBROUTINE OPTMUM(PTSET,M,N,THRESH,NCODE,FLG)
      INTEGER PTSET(110,6),P(6),D(6),ORDER(110)
      INTEGER FLG
      NCODE = 0
      DO 100 I = 1,M
100      ORDER(I) = I
       NPTS = M
       MM = 1
110    IFL = 0
      DO 500 I = 1,N
```

```
121.      500      O(I) = I
122.               IF (MM .GT. 1) GOTO 80
123.               CALL ENCODE(MM,N,PTSET,ORDER,O,NPTS,IFL,NUM,THRESH,FLG)
124.               IF (NUM .EQ. 0) GOTO 70
125.               NCODE = NUM
126.               RETURN
127.      70       MM = MM + 1
128.      80       CALL PROCES(PTSET,ORDER,P,NPTS,N)
129.               IF (FLG .EQ. 1) GO TO 30
130.               PRINT 1,(P(I),I=1,N)
131.       1       FORMAT(//,10X,'PRINTOUT OF P ARRAY IS:', 6I5)
132.      30       CALL SORT( P,O,N)
133.               IF (FLG .EQ. 1) GO TO 40
134.               PRINT 2 , (O(I),I=1,N)
135.       2       FORMAT(//,10X,'ORDER OF VARIOUS HANLES IS:', 6I5)
136.      40       CALL SURTPT(PTSET,O,ORDER,NPTS,N,M)
137.               CALL ENCODE(MM,N,PTSET,ORDER,O,NPTS,IFL,NUM,THRESH,FLG)
138.                NP = 0
139.               DO 200 I = 1,NPTS
140.                  IF(ORDER(I) .EQ. 0) GO TO 200
141.                  NP = NP + 1
142.                  ORDER(NP) = ORDER(I)
143.      200       CONTINUE
144.                NPTS = NP
145.                NCODE = NCODE + NUM
146.                IF (NPTS .EQ. 1) GO TO 20
147.                IF (NPTS .LE.0) RETURN
148.                IF (NUM .LT.1 .AND. IFL .EQ. 1) MM = MM + 1
  9.                IF (NUM .LT. 1 .AND. IFL .EQ. 0) GO TO 20
 10.                GO TO 110
151.      20        IF (FLG .EQ. 1) GO TO 50
152.                PRINT 5, MM
153.      5         FORMAT (//,10X,'HANDLE LENGTH = ',I5)
154.                PRINT 6 ,(O(I),I= 1,MM)
155.      6         FORMAT(/,10X,'HANDLES ARE :',6I5)
156.                PRINT 7
157.      7         FORMAT(/,10X,' CORRESPONDING GH CODES ARE:')
158.      50        NCODE = NCODE + NPTS
159.                IF (FLG .EQ. 1) RETURN
160.               DO 300 I = 1,NPTS
161.                 II = ORDER(I)
162.                 DO 333 J=1,N
163.                   JJ = O(J)
164.      333       P(J) = PTSET(II,JJ)
165.                PRINT 8,(P(K),K=1,MM)
166.       8        FORMAT(10X,'(',6I5,')')
167.                IF (N-MM .EQ.0) GO TO 300
168.                NI = MM+1
169.               PRINT 9,(P(K),K=NI,N)
170.      9        FORMAT('+',42X,';',5I5)
171.               PRINT 10 ,(P(K),K=NI,N)
172.      10       FORMAT('+',65X,';',5I5)
173.               PRINT 11
174.      11       FORMAT('+',85X,')')
175.      300      CONTINUE
176.               RETURN
177.               END
178.      CCCCCCCCCC
179.      C
180.      C    PROCES(PTSET,ORDER,NPTS,N):- COMPUTES THE MEASURE OF PRIORTY
```

```
181.      C           ,P(I) FOR EACH COLUMN I WHERE I=1,...,N. FOR A GIVEN
182.      C           POINT SET IN ARRAY PTSET CONTAINING NPTS NO. OF POINTS
183.      C           IN N DIMENSIONAL SPACE.
184.      C    CALLS:
185.      C     (1) HPODER:  TO COMPUTE P(I) FOR ONE PARTICULAR COLUMN I.
186.      C
187.      CCCCCCCCCC
188.            SUBROUTINE PROCES(PTSET,ORDER,P,NPTS,N)
189.            INTEGER PTSET(110,6),ORDER(110),TEMPR(110),P(6)
190.            DO 100  I = 1,N
191.            DO 200  J = 1,NPTS
192.             JJ = ORDER(J)
193.      200   TEMPR(J) = PTSET(JJ,I)
194.            CALL HPODER(I,TEMPR,NPTS,P)
195.      100   CONTINUE
196.            RETURN
197.            END
198.      CCCCCCCCCC
199.      C
200.      C  HPODER(I,TEMPR,M,P):- COMPUTES P(I) FOR A COLUMN VECTOR TEMPR
201.      C       ,IT USES HEAP STRUCTURE FOR THAT AND HENCE.
202.      C    CALLS:
203.      C     (1) REHEAP: TO MAKE HEAP OF ELEMENTS OF COLUMN VECTOR TEMPR.
204.      C
205.      CCCCCCCCCC
206.            SUBROUTINE HPODER(I,TEMPR,M,P)
207.            INTEGER TEMPR(110),P(6)
208.            INTEGER TEMP,ELMENT,PSUM,SUM
 09.             M1 = M
 10.             L = M/2 + 1
211.      10     L = L-1
212.             CALL REHEAP(L,M1,TEMPR)
213.             IF (L .GT. 1) GOTO 10
214.             ELMENT = 0
215.             PSUM = 0
216.             SUM = 0
217.      20     TEMP = TEMPR(1)
218.             TEMPR(1) = TEMPR(M1)
219.             IF (ELMENT .EQ. TEMP)GO TO 30
220.              IF(SUM .GT. 1) PSUM = PSUM + SUM * SUM
221.              ELMENT = TEMP
222.              SUM = 1
223.              GO TO 40
224.      30     SUM = SUM + 1
225.      40     M1 = M1 - 1
226.             IF(M1 .GT. 0) GO TO 50
227.               IF (SUM .GT. 1) PSUM = PSUM + SUM *SUM
228.               P(I) = PSUM
229.               RETURN
230.      50     CALL REHEAP(1,M1 TEMPR)
231.             GOTO 20
232.             END
233.      CCCCCCCCCC
234.      C  REHEAP(L,M1,TEMPR):- MAKES A HEAP OF THE ELEMENTS OF ARRAY TEMPR
235.      C       STARTING FROM TEMPR(L) TO TEMPR(M1).
236.      C
237.      CCCCCCCCCC
238.            SUBROUTINE REHEAP(L,M1,TEMPR)
239.            INTEGER TEMPR(110)
240.            INTEGER FLAG , X
```

```
241.            II = L
242.            JJ = 2*II
243.            X = TEMPR(II)
244.            FLAG = 1
245.            IF (JJ .GT. M1) RETURN
246.      40    IF(JJ .GE. M1) GOTO 10
247.            IF (TEMPR(JJ) .LT. TEMPR(JJ+1)) JJ =JJ+1
248.      10    IF(X .GE. TEMPR(JJ)) GOTO 20
249.            TEMPR(II)  =  TEMPR(JJ)
250.            II = JJ
251.            JJ = 2*II
252.            TEMPR(II) = X
253.            GOTO 30
254.      20    FLAG = 0
255.      30    IF (JJ .LE. M1 .AND. FLAG .EQ. 1) GOTO 40
256.            RETURN
257.            END
258.      CCCCCCCCCC
259.      C
260.      C  SORT(P,O,N):- SORTS P ARRAY CONTAINING N ELEMENTS AND DURING
261.      C     THIS SORT ALSO MOVES THE ELEMENTS OF ARRAY O IN THE SAME FASH-
262.      C     ION SO THAT INORDER TO CHOOSE A HANDLE SET OF M LENGTH, WE CAN
263.      C     CHOOSE FIRST M-1 HANDLES STORED IN FIRST M-1 POSITION OF O.
264.      C
265.      CCCCCCCCCC
266.            SUBROUTINE SORT(P,O,N)
267.            INTEGER O(6),P(6)
268.            INTEGER FLAG, TOP, TEMP, BOTTOM
 9.             FLAG = 1
 0.             BOTTOM = N
71.             TOP = 1
272.      10    IF(FLAG .NE. 0 .AND. TOP .LT. BOTTOM) GO TO 20
273.            RETURN
274.      20    FLAG = 0
275.            II = BOTTOM -1
276.            DO 100 I = 1,II
277.            IF (P(I) .GE. P(I+1)) GO TO 100
278.                 X   = P(I)
279.                P(I) =P(I+1)
280.                P(I+1) = X
281.                TEMP =O(I)
282.                O(I) =O(I+1)
283.                O(I+1)=TEMP
284.                FLAG = 1
285.      100    CONTINUE
286.             BOTTOM = BOTTOM -1
287.             GO TO 10
288.             END
289.      CCCCCCCCCC
290.      C
291.      C    SORTPT(PTSET,O,ORDER,M,N,M2):- TO SORT POINT IN ARRAY PTSET
292.      C      W.R.T. NEW HANDLE VECTOR IN O . ORDER CONTAINTS POINTERS TO
293.      C      POINTS IN PTSET WHICH HAVE NOT BEEN DELETED YET,M IS THE
294.      C      NUMBER OF SUCH POINTS WHILE THERE WERE M2 POINTS IN THE
295.      C      ORIGINAL POINT SET. N IS THE DIMENSION OF SPACE.
296.      C
297.      C      HEAPSORT IS USED HERE, THEREFORE:
298.      C    CALLS:
299.      C     (1) REHIP: TO MAKE HEAP.
300.      C     (2) COMPAR: TO COMPARE ORDER OF TWO POINTS.
```

```
C     (3) EXCHANG: TO EXCHANG POSITION OF TWO PCINTS IN PTSET.
C
CCCCCCCCCC
      SUBROUTINE SORTPT(PTSET,D,ORDER,M,N,M2)
      INTEGER PTSET(110,6),D(6),ORDER(110)
      M1 = M
      L = M1/2 + 1
10    L = L-1
      CALL REHIP(L,M1,M2,N,D,PTSET,ORDER)
      IF(L .GT.1) GOTO 10
20    CALL EXCHAN(ORDER(1),ORDER(M1),N,PTSET)
      M1 = M1-1
      CALL REHIP(1,M1,M2,N,D,PTSET,ORDER)
      IF(M1 .GT. 1) GO TO 20
      RETURN
      END
CCCCCCCCC
C
C    REHIP(L,M1,M2,N,D,PTSET,ORDER):- MAKES A HEAP STRUCTURE
C       OF THOSE POINTS IN PTSET WHOSE POINTER ARE IN ARRAY ORDER,
C       STARTING FROM ORDER(L) TO ORDER(M1).
C       M2 IS THE NUMBER OF POINTS IN ORIGINAL PCINTSET
C
CCCCCCCCCC
      SUBROUTINE REHIP(L,M1,M2,N,D,PTSET,ORDER)
      INTEGER PTSET(110,6),D(6),ORDER(110)
      INTEGER FLAG ,X
      II = L
      JJ = 2*II
      I = ORDER(II)
      X = M2 + 1
      DO 5 LL=1,N
5     PTSET(X,LL) = PTSET(I,LL)
      FLAG = 1
      IF(JJ .GT.M1) RETURN
40    IF(JJ .GE. M1) GO TO 10
      CALL COMPAR(ORDER(JJ),ORDER(JJ+1),N,D,PTSET,ICOMP)
      IF (ICOMP .LT.0) JJ = JJ+1
10    CALL COMPAR(X,ORDER(JJ),N,D,PTSET,ICOMP)
      IF (ICOMP .GE.0) GO TO 20
      I = ORDER(II)
      J = ORDER(JJ)
      DO 100 LL = 1,N
100   PTSET(I,LL) = PTSET(J,LL)
      II = JJ
      JJ = 2*II
      I = ORDER(II)
      DO 200 LL = 1,N
200   PTSET(I,LL) = PTSET(X,LL)
      GO TO 30
20    FLAG = 0
30    IF (JJ .LE. M1 .AND. FLAG .EQ.1) GO TO 40
      RETURN
      END
CCCCCCCCCC
C
C  COMPAR(I,J,N,D,PTSET,ICOMP):- COMPARES TWO POINTS I & J W.R.T. A
C     HANDLE VECTOR IN D, AND RETURNS IN ICOMP:
C       1 IF ITH POINT > JTH POINT.
C       0 IF ITH POINT = JTH POINT.
```

```
361.      C       -1 IF ITH POINT < JTH POINT.
362.      C
363.      CCCCCCCCCC
364.            SUBROUTINE COMPAR(I,J,N,D,PTSET,ICOMP)
365.            INTEGER PTSET(110,6),D(6)
366.            II = 1
367.      100   III = D(II)
368.            IF (PTSET(I,III) .NE. PTSET(J,III)) GO TO 10
369.            IF (II .LT. N) GO TO 5
370.            ICOMP = 0
371.            RETURN
372.      5     II = II+1
373.            GO TO 100
374.      10    IF(PTSET(I,III) .LT. PTSET(J,III)) ICOMP = -1
375.            IF (PTSET(I,III).GT.PTSET(J,III)) ICOMP = 1
376.            RETURN
377.            END
378.      CCCCCCCCC
379.      C
380.      C   EXCHAN(I,J,N,PTSET):- SWITCHES THE POSITIONS OF TWO POINTS I & J
381.      C     IN ARRAY PTSET WHICH CONTAINS A POINT SET IN N DIMENSIONAL
382.      C     SPACE.
383.      C
384.      CCCCCCCCCC
385.            SUBROUTINE EXCHAN(I,J,N,PTSET)
386.            INTEGER PTSET(110,6)
387.            DO 100 II = 1,N
388.            ITEMP = PTSET(I,II)
389.            PTSET(I,II) = PTSET(J,II)
  0.            PTSET(J,II) = ITEMP
 91.      100   CONTINUE
 92.            RETURN
 93.            END
394.      CCCCCCCCCC
395.      C
396.      C   ENCODE(M,N,PTSET,ORDER,D,NPTS,IFL,NUM,THRESH,FLG):- GENERATE
397.      C     GHM CODES FOR A POINT SET PTSET, CONTAINING NPTS POINTS IN
398.      C     N DIMENSIONAL SPACE. ONLY THOSE HYPERCUBES WHICH HAVE DESITY
399.      C     => THAN THRESHOLD DENSITY, THRESH, ARE INCLUDED IN OPTIMAL GH
400.      C     ENCODING. FOR A GIVEN M, NUM CONTAINS THE NUMBER OF SUCH
401.      C     HYPERCUBE GENERATED W.R.T. A GIVEN HANDLE VECTOR IN D.
402.      C     ORDER ARRAY CONTAINS POINTERS TO POINTS IN PTSET WHICH ARE
403.      C     STILL IN POINTSET. FLG IS A FLAG USED TO TURN OFF & ON
404.      C     PRINTING OF OPTIMAL GH ENCODED TUPLES.
405.      C   CALLS:
406.      C *   (1) MINMAX: TO FIND THE MINIMUM & MAXIMUM ELEMENT IN A
407.      C               PARTICULAR COLUMN FOR SOME PONTS CONSIDERED TO BE IN
408.      C               IN THE CURRENT HYPERCUBE.
409.      C     (2)  DENSITY: TO COMPUTE THE DENSITY OF A GENERATED HYPERCUBE.
410.      C
411.      CCCCCCCCCC
412.            SUBROUTINE ENCODE(M,N,PTSET,ORDER,D,NPTS,IFL,NUM,THRESH,FLG)
413.            INTEGER FLG,FLAG
414.            INTEGER PTSET(110,6),D(6),CODE(12),ORDER(110)
415.            MM = M -1
416.            IF (FLG .EQ. 1) GO TO 70
417.            PRINT 1
418.      1     FORMAT(//)
419.            PRINT 5,M
420.      5     FORMAT(11X,'GH',I2,' CODES ARE:')
```

```
      PRINT 6,(O(I),I=1,MM)
6     FORMAT('0',10X,'HANDLES ARE:',615)
70    IF(MM .EQ. N) GO TO 20
      IF (MM .EQ. 0) GO TO 10
      NUM = 0
      IFL = 0
      M1 = NPTS +1
      I = 1
      J = 1
      DO 100 K = 2,M1
        IF (K .GT. NPTS) GO TO 40
        FLAG = 0
        I1 = ORDER(K)
        I2 = ORDER(K-1)
        DO 200 JI = 1,MM
          JJ = O(JI)
          IF(PTSET(I1,JJ) .EQ. PTSET(I2,JJ)) GO TO 200
          FLAG = 1
          GO TO 330
200     CONTINUE
330     IF (FLAG .EQ. 1) GO TO 40
        J = J + 1
        GO TO 100
40      II = ORDER(I)
        DO 300 JI = 1,MM
          JJ = O(JI)
          CODE(JI) = PTSET(II,JJ)
300     CONTINUE
        N1 = MM+1
        L = MM
        NN = N -MM
        DO 400 JI = N1,N
          JJ = O(JI)
          CALL MINMAX(PTSET,ORDER,JJ,1,J,MIN,MAX)
          L = L + 1
          CODE(L) = MIN
400       CODE(L+NN) = MAX
        DEN = DENSTY(CODE,MM,N,I,J)
        IF(DEN .GE.THRESH) GO TO 111
        IFL = 1
        GO TO 555
111     IF (I .EQ. J) GO TO 555
          N2 = L
          N3 = L + NN
        DO 333 IK = I,J
333       ORDER(IK) = 0
        IF ( FLG .EQ. 1) GO TO 72
          PRINT 32,(CODE(I),I = 1,MM)
32        FORMAT('0',10X,'(',613)
          PRINT 33,(CODE(I),I=N1,N2)
33        FORMAT('+',30X,',',613)
          N22 = N2+1
          PRINT 34,(CODE(I),I=N22,N3)
34        FORMAT('+',50X,',',613)
          PRINT 35
35        FORMAT('+',60X,')')
72        NUM = NUM + 1
555       I = K
          J = K
100     CONTINUE
```

```
      CCCCCCCCCC
      C
      C
            SUBROUTINE EXHAUS(PTSET,M,N,NCODE,IFL)
            INTEGER PTSET(110,6),O(6),TABLE(720,6),NCODE(5,3)
            CALL PRCOMB(TABLE,N)
            ITERM = 1
            DO 100 I = 1,N
  100       ITERM = ITERM*I
            DO 200 I = 1,ITERM
               DO 300 J= 1,N
  300          O(J) = TABLE(I,J)
               IF(IFL .EQ. 1) GO TO 88
               PRINT 1
    1          FORMAT(//,11X,'HANDLES CHOSEN ARE:')
               PRINT 2,(O(JI,JI=1,N)
    2          FORMAT('0',10X,415)
   88          CALL SORTPT(PTSET,O,M,N)
               IF(IFL .EQ. 1) GO TO 89
               PRINT 3
    3          FORMAT(//,11X,'FOLLOWING ARE GH CODES W.R.T. ABOVE '
     $            ,'HANDLES:')
   89          NI = N + 1
               DO 500 J = 1,NI
                 CALL ENCODE(J,M,N,O,PTSET,NUM,IFL)
                  IF(J .LE.1 .OR. J .GE.NI) GO TO 500
                  IF(NUM .LT. NCODE(J-1,2)) NCODE(J-1,2) = NUM
                  IF(NUM .GT. NCODE(J-1,3)) NCODE(J-1,3) = NUM
  500           CONTINUE
                IF( IFL .EQ. 1) GO TO 200
               PRINT 4
    4          FORMAT(//,100('.'))
  200          CONTINUE
               RETURN
               END
      CCCCCCCCCC
      C
      C     PRCOMB(TABLE,N): TO FIND ALL POSSIBLE SET OF HANDLES, USING
      C        PERMUTATION COMBINATION.EACH ROW OF TABLE CONTAINS ON SET
      C        HANDLES.N IS THE DIMENSION OF SPACE.
      C    CALLS:
      C      (1) FILL: TO STORE PROPER COLUMN NUMBERES IN TABLE.
      C
      CCCCCCCCCC
            SUBROUTINE PRCOMB(TABLE,N)
            INTEGER TABLE(720,6)
            INTEGER COL
            NUM = N
            DO 100 I = 1,N
  100       TABLE(1,I) = I
            K = 1
            COL = 1
   10       L = 1
            NI = NUM - 1
            ITERM = 1
            DO 200 I = 1,NI
  200       ITERM = ITERM *I
            DO 300 KI = 1,K
               DO 400 I = 1,NUM
                  II = L
```